Application of an integral approach to the parallel algorithm of 3D wave fields simulation in generalized coordinates

PAVEL TITOV, DMITRIY WEINS & IGOR CHERNYKH

ICMMG SB RAS, Novosibirsk

Introduction

For the past few years, exascale and big data are at the very center of interest of supercomputer community. One can get familiar with it [1]-[4]. While the architecture of exaflop supercomputer is still being discussed, one thing is clear: it is how necessary to develop algorithms and software for such systems now. The software should be able to work effectively with hundreds of thousands, even millions of processors, and to store large amounts of data.

This is where an integral approach to algorithm development comes in. We consider an integral approach to be a sum of elements listed below:

1. The co-design: an adaptation of both the algorithm and the mathematical method to the supercomputer architecture at every stage of the problem solution.

2. Energy efficiency of the algorithm: analysis of energy consumption, efficiency of the memory and the network resources usage, as well as the computation cores load.

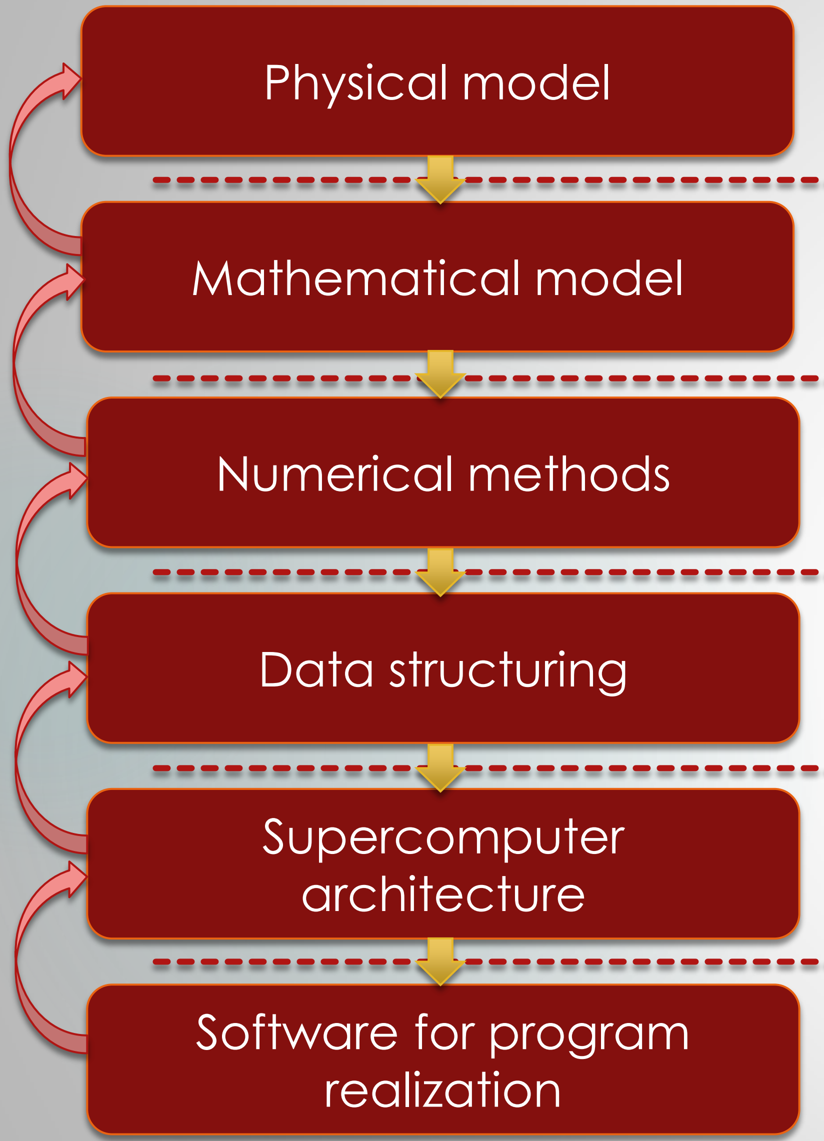
3. Simulation modeling: development of the preemptive algorithms and software based on simulation tests on different architectures.

An integral approach to creating algorithms and software for high performance computing (HPC) is being developed at the ICMMG SB RAS. It was successfully applied for problems of astrophysics [5] and geophysics [6]. In these papers the system AGNES [7] was used for simulation of the algorithm performance on massive parallel systems.

The difference of this paper from [5,6] is that we utilize Erlang istead [8]. As [9] shows, on larger number of cores (1 million cores and higher) the Erlang provides more reliable results than the AGNES.

It is first time that an integral approach and the simulation tools of Erlang would be applied for a geophysical problem stated in generalized coordinates. Earlier generalized coordinates for the wave field modeling were used in the papers [10,11] for 2D case, as well as in the papers of one of co-authors [12,13] for 2D case and [14] for 3D case. In [14] also the co-design concept was applied.

Co-design of the parallel algorithm



Physical model

Physical problem statement

Mathematical model

Mathematical statement of the physical problem

Numerical methods

Development of the numerical method to solve mathematical problem

Data structuring

Choosing the structure of the data and parallel algorithm

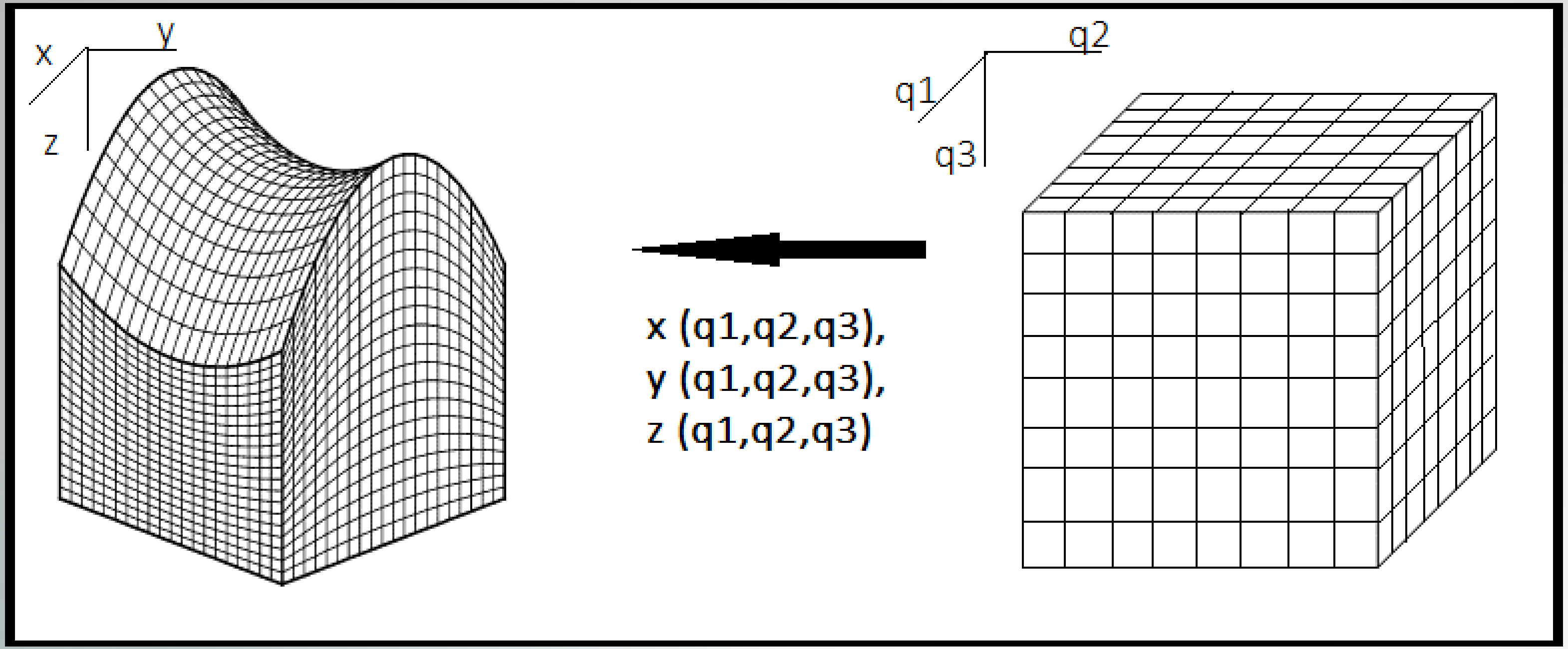
Supercomputer architecture

Consideration of supercomputer structure and network parameters

Software for program realization

Utilization of software necessary to create a parallel program version of the algorithm

Curvilinear mesh construction



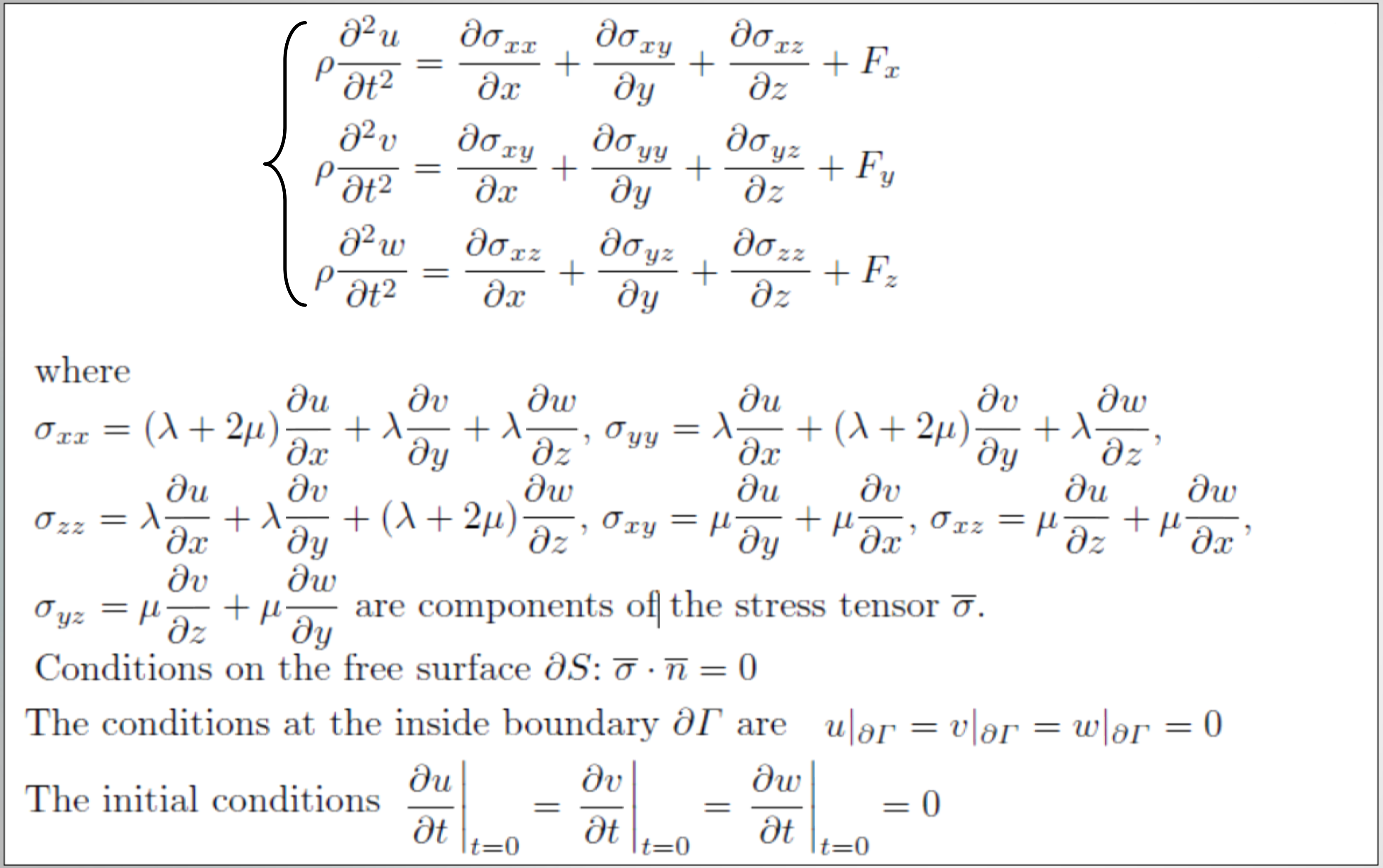
The technique of constructing a 3D mesh is considered in [22, * Titov]. In this study we will just note an important point: near the free surface, all the coordinate lines of the curvilinear mesh are mutually orthogonal.

The main advantage of building a mesh with this method is that grid nodes are calculated analytically and therefore, it is not time consuming and easy to program.

Local orthogonality allows better free surface condition approximation.

* Titov P.: Modeling of elastic waves in media with a complex free surface topography. Vestnik of NSU: Information Technologies, vol. 16, c. 4, 2018; pp. 153-166. (In Russian)

Problems statement in Cartesian and generalized coordinates



$$\begin{cases} \rho\dfrac{\partial^2 u}{\partial t^2} = \dfrac{\partial \sigma_{xx}}{\partial x} + \dfrac{\partial \sigma_{xy}}{\partial y} + \dfrac{\partial \sigma_{xz}}{\partial z} + F_x \\ \rho\dfrac{\partial^2 v}{\partial t^2} = \dfrac{\partial \sigma_{xy}}{\partial x} + \dfrac{\partial \sigma_{yy}}{\partial y} + \dfrac{\partial \sigma_{yz}}{\partial z} + F_y \\ \rho\dfrac{\partial^2 w}{\partial t^2} = \dfrac{\partial \sigma_{xz}}{\partial x} + \dfrac{\partial \sigma_{yz}}{\partial y} + \dfrac{\partial \sigma_{zz}}{\partial z} + F_z \end{cases}$$

where

$\sigma_{xx} = (\lambda + 2\mu)\dfrac{\partial u}{\partial x} + \lambda\dfrac{\partial v}{\partial y} + \lambda\dfrac{\partial w}{\partial z}$, $\sigma_{yy} = \lambda\dfrac{\partial u}{\partial x} + (\lambda + 2\mu)\dfrac{\partial v}{\partial y} + \lambda\dfrac{\partial w}{\partial z}$,

$\sigma_{zz} = \lambda\dfrac{\partial u}{\partial x} + \lambda\dfrac{\partial v}{\partial y} + (\lambda + 2\mu)\dfrac{\partial w}{\partial z}$, $\sigma_{xy} = \mu\dfrac{\partial u}{\partial y} + \mu\dfrac{\partial v}{\partial x}$, $\sigma_{xz} = \mu\dfrac{\partial u}{\partial z} + \mu\dfrac{\partial w}{\partial x}$,

$\sigma_{yz} = \mu\dfrac{\partial v}{\partial z} + \mu\dfrac{\partial w}{\partial y}$ are components of the stress tensor $\overline{\sigma}$.

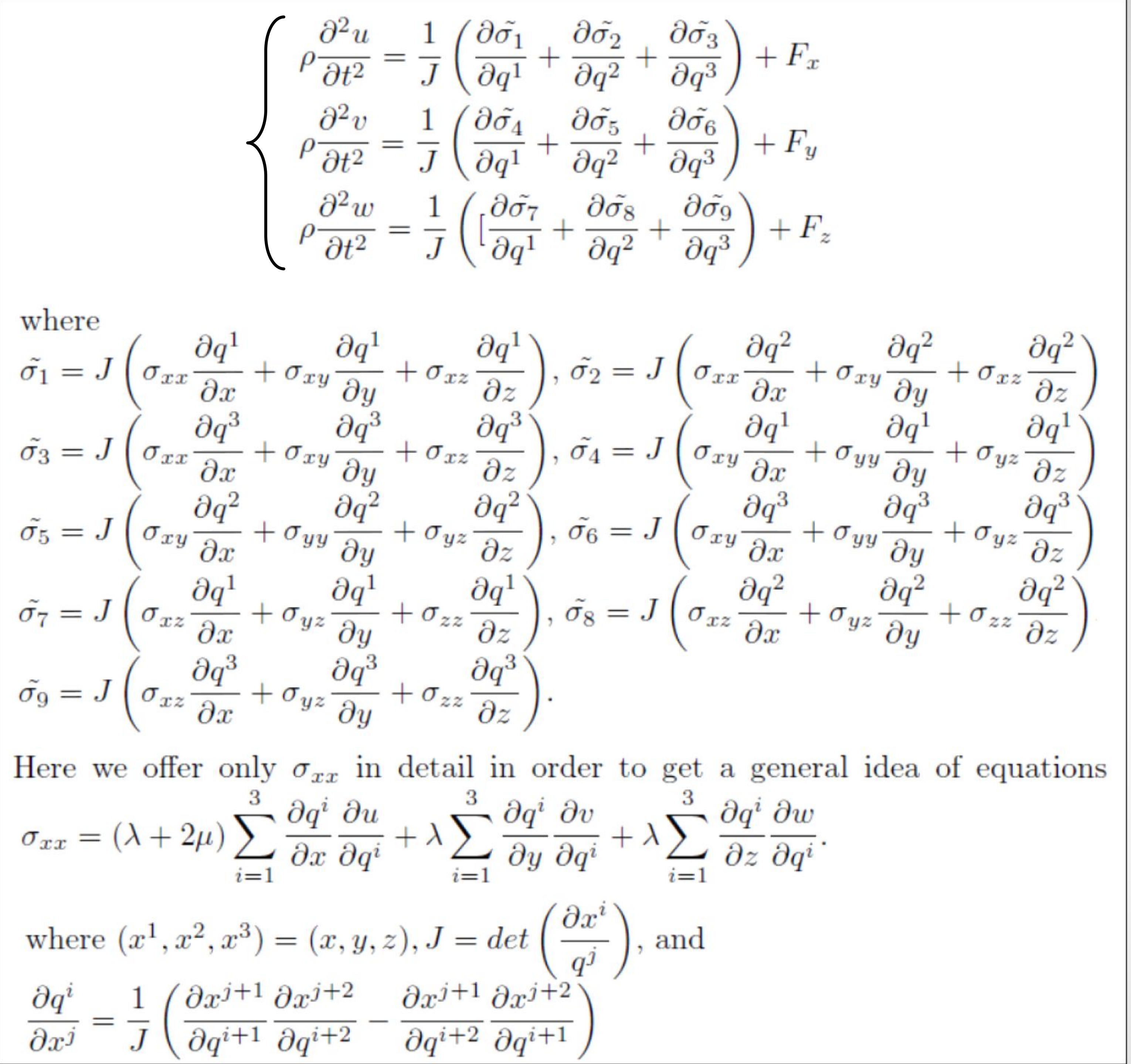
Conditions on the free surface ∂S: $\overline{\sigma} \cdot \overline{n} = 0$

The conditions at the inside boundary $\partial\Gamma$ are $u|_{\partial\Gamma} = v|_{\partial\Gamma} = w|_{\partial\Gamma} = 0$

The initial conditions $\left.\dfrac{\partial u}{\partial t}\right|_{t=0} = \left.\dfrac{\partial v}{\partial t}\right|_{t=0} = \left.\dfrac{\partial w}{\partial t}\right|_{t=0} = 0$

The wave field simulation is carried out based on the numerical solution of the linear system of elasticity theory, expressed via displacements in the Cartesian coordinates.

We consider (q^1, q^2, q^3) to be the new generalized coordinates. Initial system must be transformed accordingly.



$$\begin{cases} \rho\dfrac{\partial^2 u}{\partial t^2} = \dfrac{1}{J}\left(\dfrac{\partial\tilde{\sigma_1}}{\partial q^1} + \dfrac{\partial\tilde{\sigma_2}}{\partial q^2} + \dfrac{\partial\tilde{\sigma_3}}{\partial q^3}\right) + F_x \\ \rho\dfrac{\partial^2 v}{\partial t^2} = \dfrac{1}{J}\left(\dfrac{\partial\tilde{\sigma_4}}{\partial q^1} + \dfrac{\partial\tilde{\sigma_5}}{\partial q^2} + \dfrac{\partial\tilde{\sigma_6}}{\partial q^3}\right) + F_y \\ \rho\dfrac{\partial^2 w}{\partial t^2} = \dfrac{1}{J}\left([\dfrac{\partial\tilde{\sigma_7}}{\partial q^1} + \dfrac{\partial\tilde{\sigma_8}}{\partial q^2} + \dfrac{\partial\tilde{\sigma_9}}{\partial q^3}\right) + F_z \end{cases}$$

where

$$\tilde{\sigma_1} = J\left(\sigma_{xx}\frac{\partial q^1}{\partial x} + \sigma_{xy}\frac{\partial q^1}{\partial y} + \sigma_{xz}\frac{\partial q^1}{\partial z}\right),\ \tilde{\sigma_2} = J\left(\sigma_{xx}\frac{\partial q^2}{\partial x} + \sigma_{xy}\frac{\partial q^2}{\partial y} + \sigma_{xz}\frac{\partial q^2}{\partial z}\right)$$

$$\tilde{\sigma_3} = J\left(\sigma_{xx}\frac{\partial q^3}{\partial x} + \sigma_{xy}\frac{\partial q^3}{\partial y} + \sigma_{xz}\frac{\partial q^3}{\partial z}\right),\ \tilde{\sigma_4} = J\left(\sigma_{xy}\frac{\partial q^1}{\partial x} + \sigma_{yy}\frac{\partial q^1}{\partial y} + \sigma_{yz}\frac{\partial q^1}{\partial z}\right)$$

$$\tilde{\sigma_5} = J\left(\sigma_{xy}\frac{\partial q^2}{\partial x} + \sigma_{yy}\frac{\partial q^2}{\partial y} + \sigma_{yz}\frac{\partial q^2}{\partial z}\right),\ \tilde{\sigma_6} = J\left(\sigma_{xy}\frac{\partial q^3}{\partial x} + \sigma_{yy}\frac{\partial q^3}{\partial y} + \sigma_{yz}\frac{\partial q^3}{\partial z}\right)$$

$$\tilde{\sigma_7} = J\left(\sigma_{xz}\frac{\partial q^1}{\partial x} + \sigma_{yz}\frac{\partial q^1}{\partial y} + \sigma_{zz}\frac{\partial q^1}{\partial z}\right),\ \tilde{\sigma_8} = J\left(\sigma_{xz}\frac{\partial q^2}{\partial x} + \sigma_{yz}\frac{\partial q^2}{\partial y} + \sigma_{zz}\frac{\partial q^2}{\partial z}\right)$$

$$\tilde{\sigma_9} = J\left(\sigma_{xz}\frac{\partial q^3}{\partial x} + \sigma_{yz}\frac{\partial q^3}{\partial y} + \sigma_{zz}\frac{\partial q^3}{\partial z}\right).$$

Here we offer only σ_{xx} in detail in order to get a general idea of equations

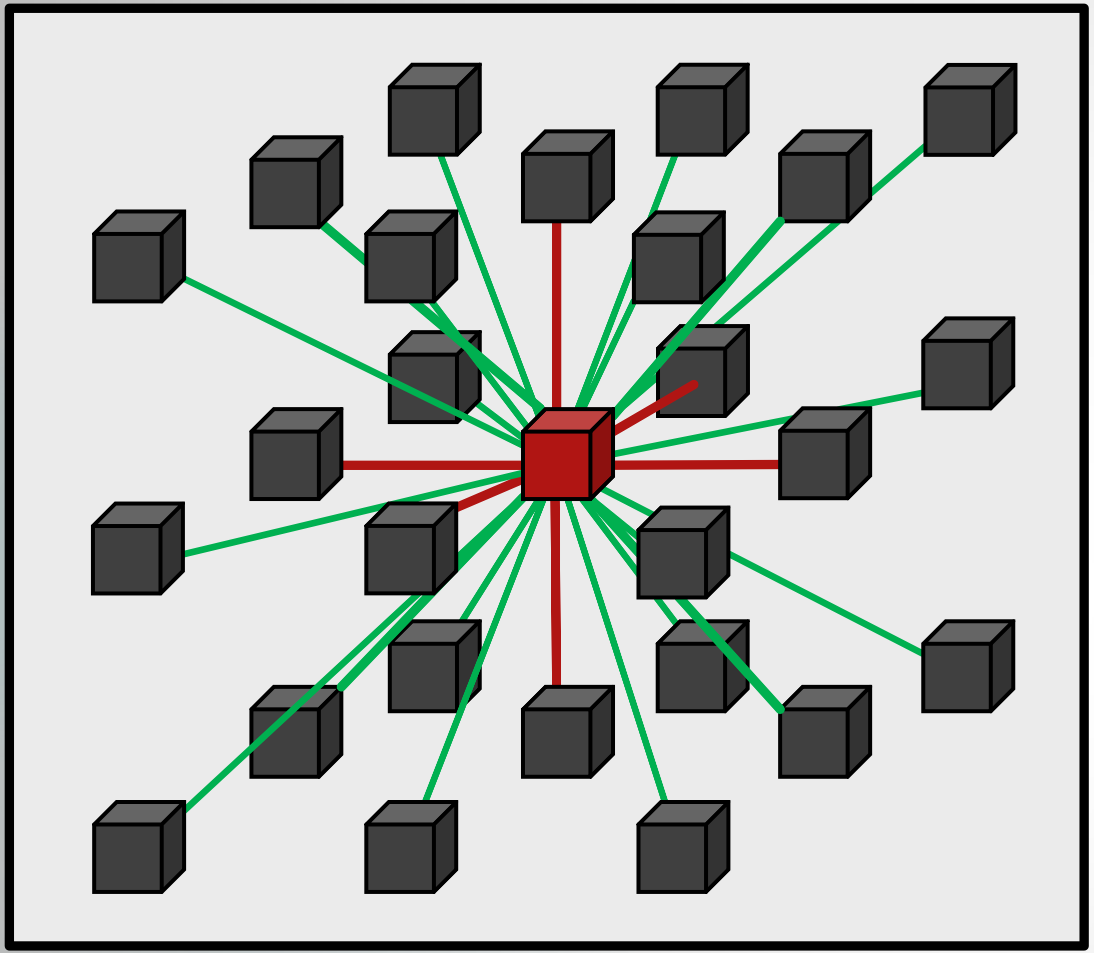
$$\sigma_{xx} = (\lambda + 2\mu)\sum_{i=1}^{3}\frac{\partial q^i}{\partial x}\frac{\partial u}{\partial q^i} + \lambda\sum_{i=1}^{3}\frac{\partial q^i}{\partial y}\frac{\partial v}{\partial q^i} + \lambda\sum_{i=1}^{3}\frac{\partial q^i}{\partial z}\frac{\partial w}{\partial q^i}.$$

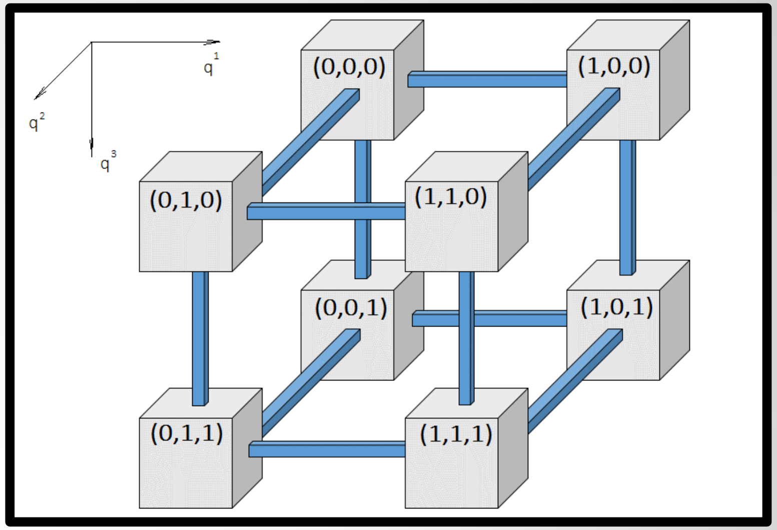
where $(x^1, x^2, x^3) = (x, y, z)$, $J = det\left(\dfrac{\partial x^i}{q^j}\right)$, and

$$\frac{\partial q^i}{\partial x^j} = \frac{1}{J}\left(\frac{\partial x^{j+1}}{\partial q^{i+1}}\frac{\partial x^{j+2}}{\partial q^{i+2}} - \frac{\partial x^{j+1}}{\partial q^{i+2}}\frac{\partial x^{j+2}}{\partial q^{i+1}}\right)$$

Equations in generalized coordinates

Neighboring processes and communication topology

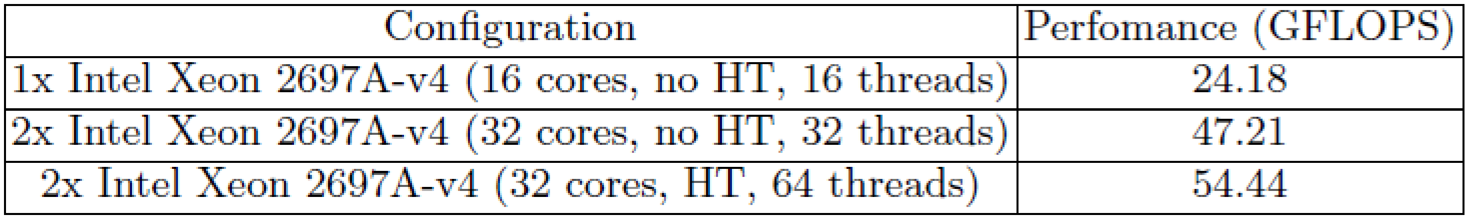




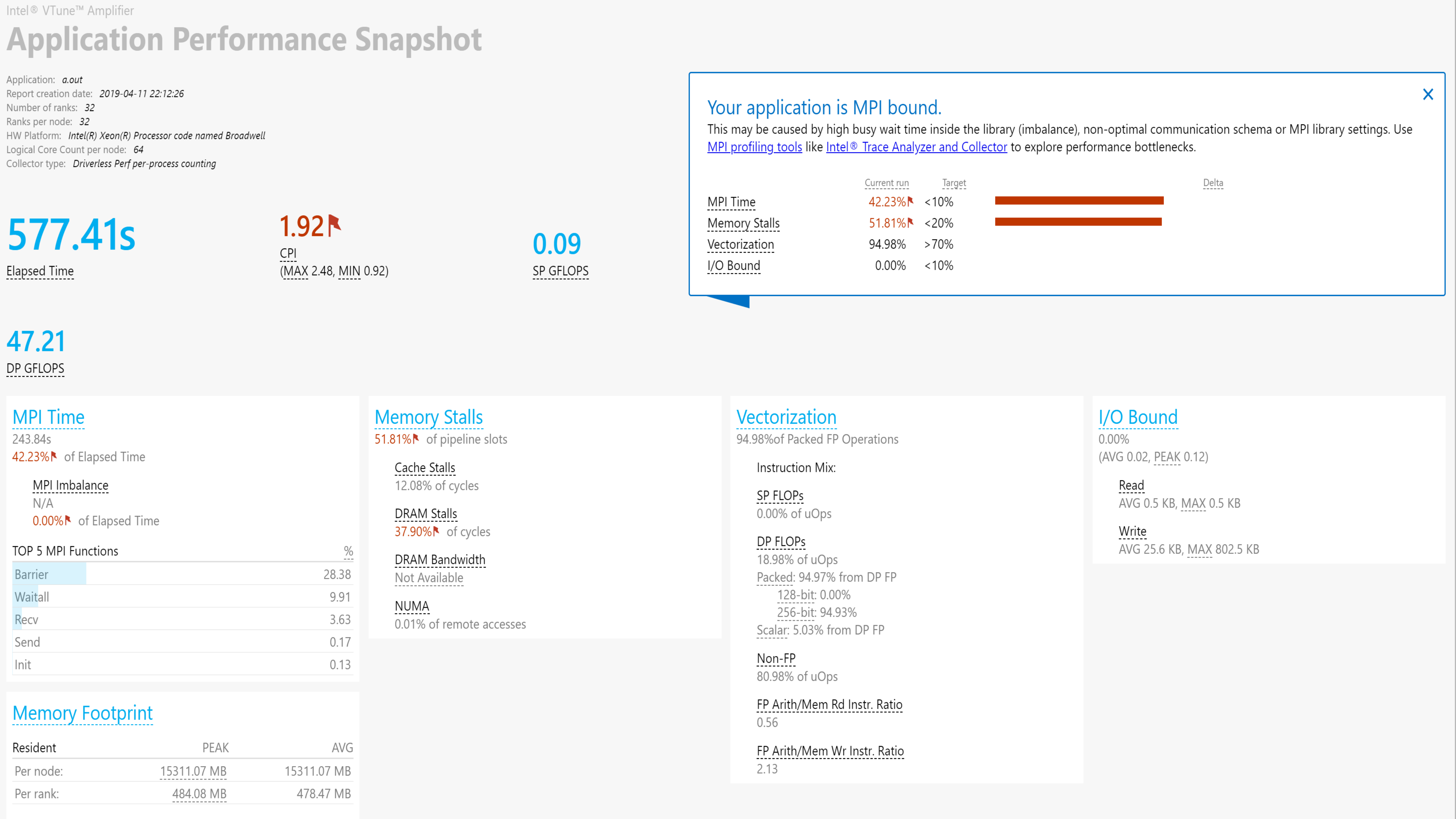
In general, we decompose the domain into small 3D-cubes, every one of which is being assigned to a single process to realize the finite difference scheme. After every time step, the neighboring processes conduct the data exchange via the created 3D-cube topology. Each process has 26 neighbors. The parallel program was developed by means of Fortran language and MPI library.

Performance analysis and energy efficiency

When we are talking about efficient employing of a massively parallel systems, we should touch on "energy efficiency". In this paper, this means the most efficient use of given computational resources and minimization of communications between computational cores that do not share the same RAM. The workload on the cores should be balanced and idle standing time minimized. The most energy effective algorithms have the highest FLOPS per Watts (Joules/sec) values. For test the NKS-1P computational node of SSCC SB RAS (2x Intel Xeon E5-2697A-v4 CPUs and 256GB of DDR4 memory) was used. For performance analysis, we used the Application Performance Snapshot of Intel Vtune Amplifier 2019 software and the Intel Fortran Compiler for code compilation and linking with O3 optimization parameter. For this code, we achieved 89% with hyper-threading and 95% without hyper-threading AVX-2 auto-vectorization of double precision floating point operations. Floating point operations per memory read instructions ratio is 0.5 and floating point operations per memory write instructions ratio is 2.1. We have no I/O operations for our code. The double precision operations performance of the code is presented in the Table below. Hyper-threading on 2 CPUs gave 13% speedup of the code. We achieved 10-15% of peak performance for this type of CPU with AVX-2 instructions. This code has 2 main bottlenecks: memory stalls and the MPI Barrier operations. The MPI Barrier operations take 26% of the computational time for 2xCPU configuration. In future work, we will change the algorithm to minimize the MPI operations time and will remove memory stalls. Due to the bottlenecks, the energy efficiency is 0.19GFLOPS/W with peak 1.5-1.7 GFLOPS/W for this type of CPU with using of AVX-2 instructions.



Configuration	Perfomance (GFLOPS)
1x Intel Xeon 2697A-v4 (16 cores, no HT, 16 threads)	24.18
2x Intel Xeon 2697A-v4 (32 cores, no HT, 32 threads)	47.21
2x Intel Xeon 2697A-v4 (32 cores, HT, 64 threads)	54.44



Intel® VTune™ Amplifier

Application Performance Snapshot

Application: *a.out*
Report creation date: *2019-04-11 22:12:26*
Number of ranks: *32*
Ranks per node: *32*
HW Platform: *Intel(R) Xeon(R) Processor code named Broadwell*
Logical Core Count per node: *64*
Collector type: *Driverless Perf per-process counting*

577.41s
Elapsed Time

1.92 ⚑
CPI
(MAX 2.48, MIN 0.92)

0.09
SP GFLOPS

47.21
DP GFLOPS

Your application is MPI bound.

This may be caused by high busy wait time inside the library (imbalance), non-optimal communication schema or MPI library settings. Use MPI profiling tools like Intel® Trace Analyzer and Collector to explore performance bottlenecks.

	Current run	Target	Delta
MPI Time	42.23% ⚑	<10%	▬
Memory Stalls	51.81% ⚑	<20%	▬
Vectorization	94.98%	>70%	
I/O Bound	0.00%	<10%	

MPI Time

243.84s
42.23% ⚑ of Elapsed Time

MPI Imbalance
N/A
0.00% ⚑ of Elapsed Time

TOP 5 MPI Functions	%
Barrier	28.38
Waitall	9.91
Recv	3.63
Send	0.17
Init	0.13

Memory Footprint

Resident	PEAK	AVG
Per node:	15311.07 MB	15311.07 MB
Per rank:	484.08 MB	478.47 MB

Memory Stalls

51.81% ⚑ of pipeline slots

Cache Stalls
12.08% of cycles

DRAM Stalls
37.90% ⚑ of cycles

DRAM Bandwidth
Not Available

NUMA
0.01% of remote accesses

Vectorization

94.98%of Packed FP Operations

Instruction Mix:

SP FLOPs
0.00% of uOps

DP FLOPs
18.98% of uOps
Packed: 94.97% from DP FP
128-bit: 0.00%
256-bit: 94.93%
Scalar: 5.03% from DP FP

Non-FP
80.98% of uOps

FP Arith/Mem Rd Instr. Ratio
0.56

FP Arith/Mem Wr Instr. Ratio
2.13

I/O Bound

0.00%
(AVG 0.02, PEAK 0.12)

Read
AVG 0.5 KB, MAX 0.5 KB

Write
AVG 25.6 KB, MAX 802.5 KB

Simulation modeling

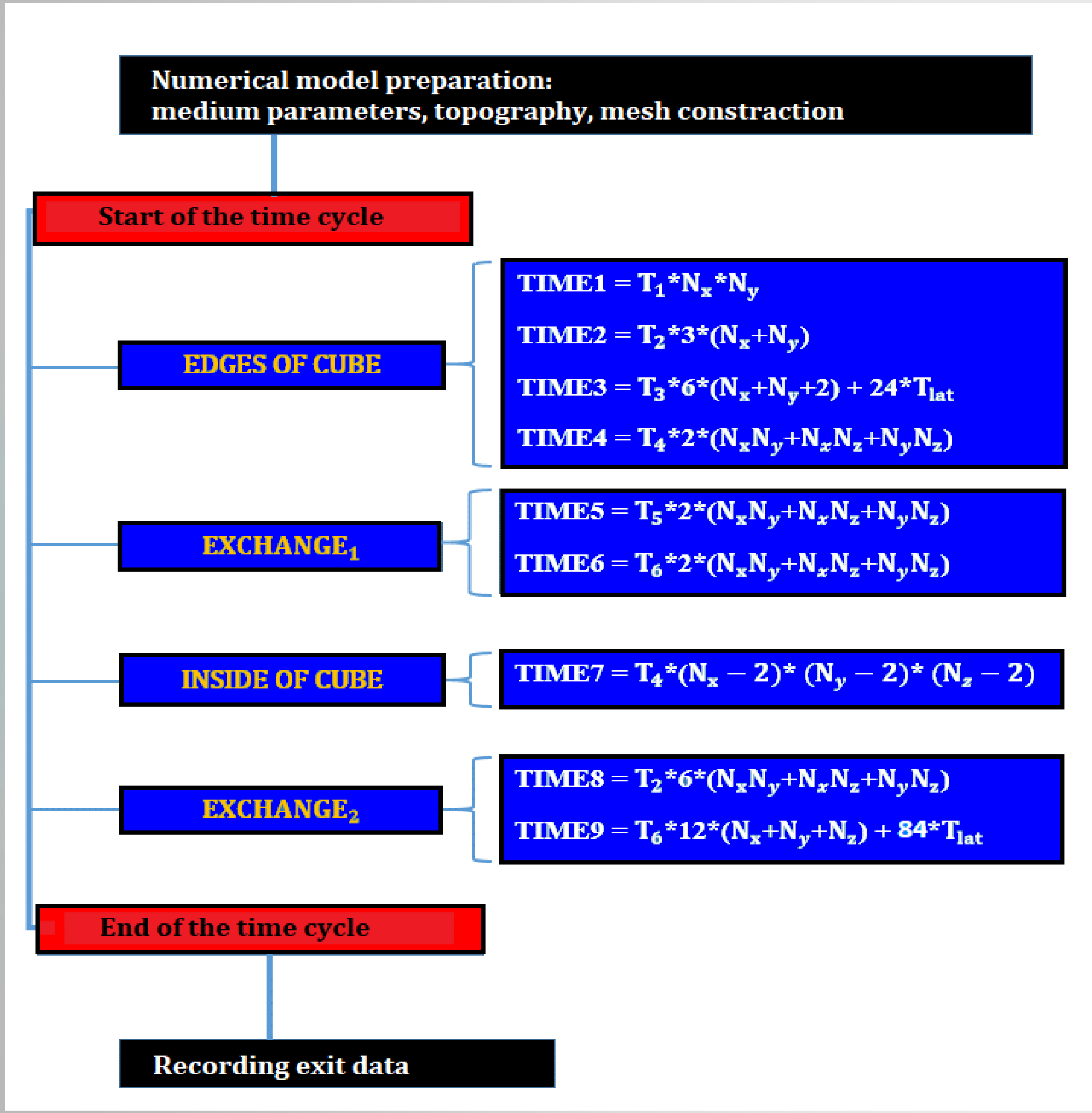
Distributed simulation based on the transmission of messages is best suited for imitation of distributed systems. To simulate the execution of parallel programs on a large number of cores, a multi-agent approach was used [* Wooldridge]. To minimize the overhead load in the communication of the agents, the Actor model was utilized [**Weins].

The program can be presented as a time function, that depends on input parameters:

- the scheme of calculations in the program;
- how long does it take to calculate one element at every time step and how many elements there is in the domain of interest;
- how many time steps does it take to execute the program;
- network parameters such as a topology and a latency of communications.
  Knowing these parameters we can simulate the program performance on a large multi-core system (hundreds of thousands and even millions of threads).

* I. Wooldridge, M.: Introduction to MultiAgent Systems England. JOHN WILEY and SONS, LTD, 2002.
** Weins Dmitry, Glinskiy Boris, Chernykh Igor: Analysis of Means of Simulation Modeling of Parallel Algorithms. Russian Supercomputer Days , Proceedings, 2018; pp. 64-75.

Algorithm representation as a time function



Numerical model preparation:
medium parameters, topography, mesh constraction

Start of the time cycle

EDGES OF CUBE

$TIME1 = T_1*N_x*N_y$

$TIME2 = T_2*3*(N_x+N_y)$

$TIME3 = T_3*6*(N_x+N_y+2) + 24*T_{lat}$

$TIME4 = T_4*2*(N_xN_y+N_xN_z+N_yN_z)$

$EXCHANGE_1$

$TIME5 = T_5*2*(N_xN_y+N_xN_z+N_yN_z)$

$TIME6 = T_6*2*(N_xN_y+N_xN_z+N_yN_z)$

INSIDE OF CUBE

$TIME7 = T_4*(N_x - 2)* (N_y - 2)* (N_z - 2)$

$EXCHANGE_2$

$TIME8 = T_2*6*(N_xN_y+N_xN_z+N_yN_z)$

$TIME9 = T_6*12*(N_x+N_y+N_z) + 84*T_{lat}$

End of the time cycle

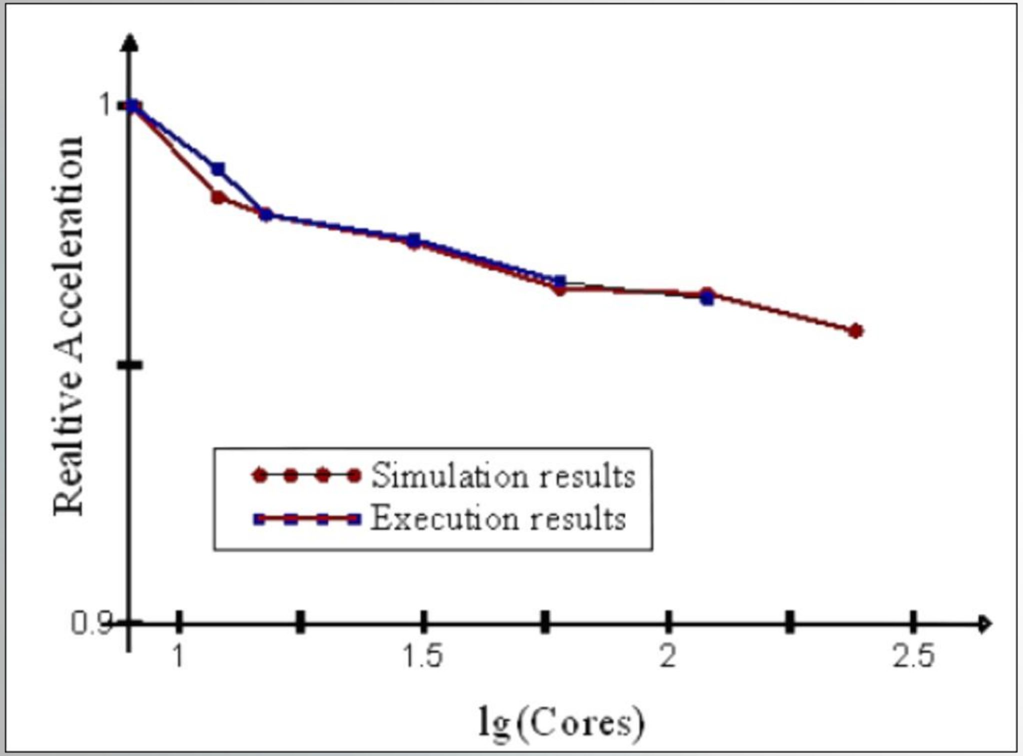
Recording exit data

The algorithm can be presented as set of time functions (**TIME1**-**TIME9**). Every part of it depends on a number of elements and the time it takes to calculate one element.

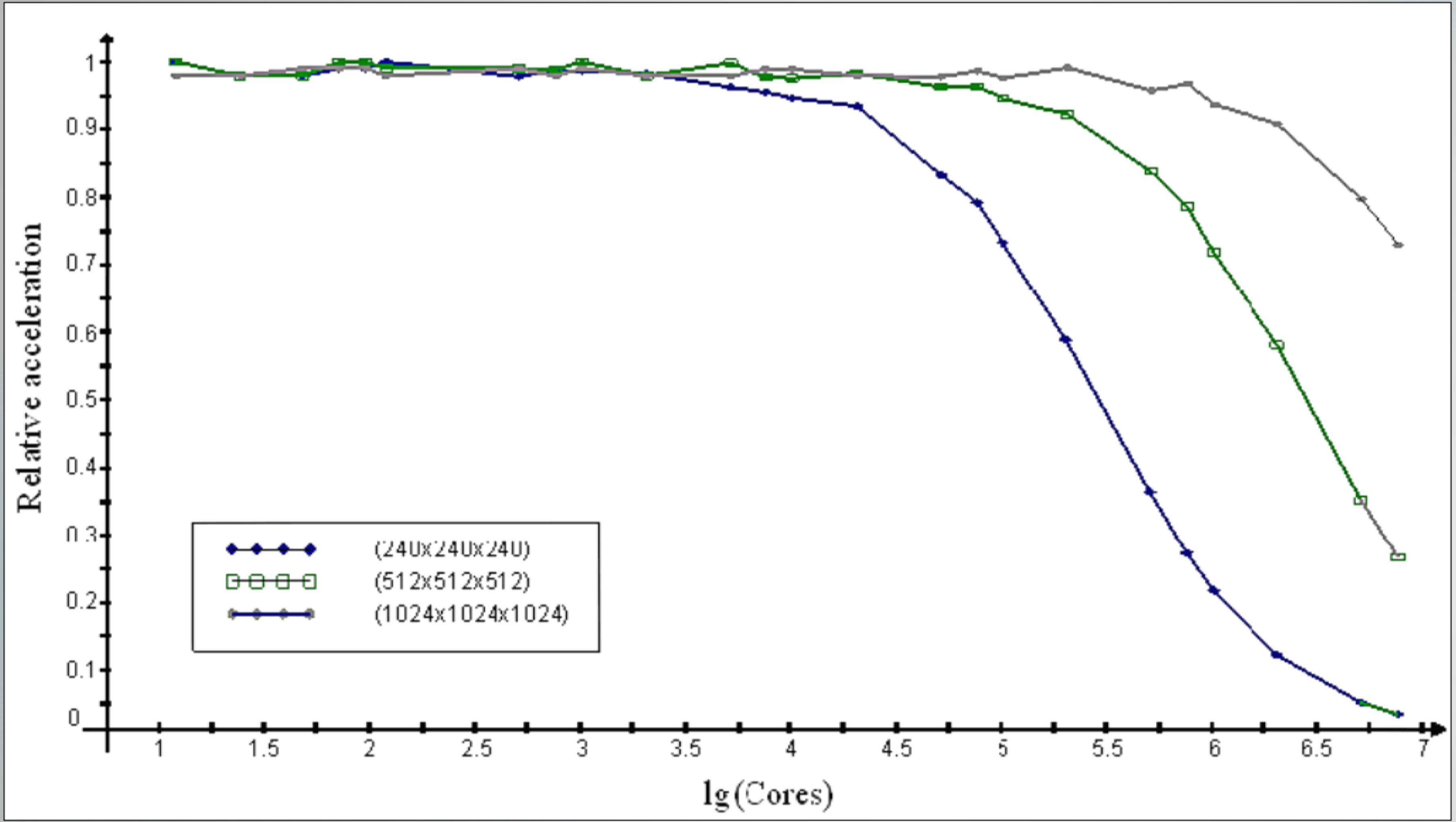
Each process gets to calculate its N_xxN_yxN_z cube of data. **T1**-**T6** are times the it takes to carry out calculations on one element for different parts of the algorithm. T_{lat} is an average latency of the network communications for given supercomputer architecture. I our case it was

Results of simulation tests via means of Erlang

The cube of 240x240x240 cells is assigned to every 12 processes. Hyper threading technology is off.
Tests are conducted on G7 servers (2*6 cores Intel Xeon processors, 48 Gb RAM) of NKS-30T cluster of SSCC.



Using simulation tools of Erlang, we can obtain evaluation of our algorithm performance on a massive parallel system.


Relative acceleration
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
1.5
2
2.5
3
3.5
4
4.5
5
5.5
6
6.5
7
lg(Cores)
(240x240x240)
(512x512x512)
(1024x1024x1024)

Conclusion

- An integral approach to developing of the algorithm and its program realization is described.

- Parallel 3D algorithm of wave field modeling in media with complex topography is presented.

- The algorithm was analyzed via means of Intel software, further recommendations on performance optimization are obtained.

- Scalability of the algorithm on massively parallel systems studied via means of simulation modeling with the aid of Erlang.

References

1. Reed, D.A., Dongarra, J.: Exascale computing and big data. Comm. ACM 58(7), 2015; pp. 56–68.

2. Dongarra, J.J., et al.: The international exascale software project roadmap. Int. J. High Perf.

Comp. App. 25(1), 2011; pp. 3–60.

3. Tabbal, A., Anderson, M., Brodowicz, M., Kaiser, H., Sterling, T.: Preliminary design

examination of the parallex system from a software and hardware perspective. Sigmetrics

Perform. Eval. Rev. 38(4), 2011; pp. 81–87.

4. Sterling, T.: Achieving scalability in the presence of asynchrony for exascale computing.

Adv. Parall. Comp. 24, 2013; pp. 104–117.

5. Boris Glinsky, Igor Kulikov, Igor Chernykh, Dmitry Weins, Alexey Snytnikov, Vladislav Nenashev, Andrey Andreev, Vitaly Egunov, Egor Kharkov:

The Co-design of Astrophysical Code for Massively Parallel Supercomputers. ICA3PP 2016 Workshops, LNCS 10049, 2016; pp. 342–353.

6. Boris Glinskiy, Igor Kulikov, Igor Chernykh, Alexey Snytnikov, Anna Sapetina, Dmitry Weins:

The Integrated Approach to Solving Large-Size Physical Problems on Supercomputers. Russian Supercomputer Days, Proceedings, 2017; pp. 278-289.

7. Podkorytov, D., Rodionov, A., Sokolova, O., Yurgenson, A.: Using agent-oriented simulation system AGNES for evaluation of sensor networks. In: Vinel, A., Bellalta, B.,

Sacchi, C., Lyakhov, A., Telek, M., Oliver, M. (eds.) MACOM 2010. LNCS, vol. 6235, pp. 247–250.

8. Cesarini F., Thompson S.: Erlang Programming. O'Reilly Media, 2009. pp. 498, ISBN-10: 0596518188 ISBN-13: 978-0596518189.

9. Weins Dmitry, Glinskiy Boris, Chernykh Igor: Analysis of Means of Simulation Modeling of Parallel Algorithms. Russian Supercomputer Days , Proceedings, 2018; pp. 64-75.

10. Daniel Appelo, N. Anders Petersson: A Stable Finite Difference method for the Elastic wave equation on complex geometries with free surfaces.

Commun. Comput. Phys. Vol.5, No. 1, 2009; pp. 84-107.

11. Jose M. Carcione: The wave equation in generalized coordinates. GEOPHYSICS, VOL. 59, NO. 12, 1994; pp. 1911-1919.

12. Titov P.: An Algorithm and a program for simulation of 2D-wave fields in areas with a curved free surface.

Materials of the conference "Scientific service on the Internet - 2014" Novorossiysk, Abrau-Dyurso, September 21-26, 2014; pp.446-455. (In Russian)

13. Titov P.: Modeling of elastic waves in media with a complex free surface topography. Vestnik of NSU: Information Technologies, vol. 16, c. 4, 2018; pp. 153-166. (In Russian)

14. Titov P.: A technology of modeling of elastic waves propagation in media with complex 3D geometry of the surface with the aid of high performance cluster. Russian Supercomputing Days, Proc., 2016, pp. 1020-1031.

15. Sapetina A.: Comparing efficiency of using different mathematical statements of a problem for 3D simulation of seismic wave field at supercomputer. Russian Supercomputing Days, Proceedings, 2018; pp. 985-995.

16. Liseykin V.: Differential meshes. Theory and applications. SB RAS Publ., 2014; 254 pp. (In Russian)

17. I. Wooldridge, M.: Introduction to MultiAgent Systems England. JOHN WILEY and SONS, LTD, 2002.

Thank you for your attention